5 - Normals Theory

In this Chapter:

- *Defining Normals*
- *Normals Calulation*
- *Detaching Faces*
- *Projection*
- *Mixing Methods*
- *Tricks*

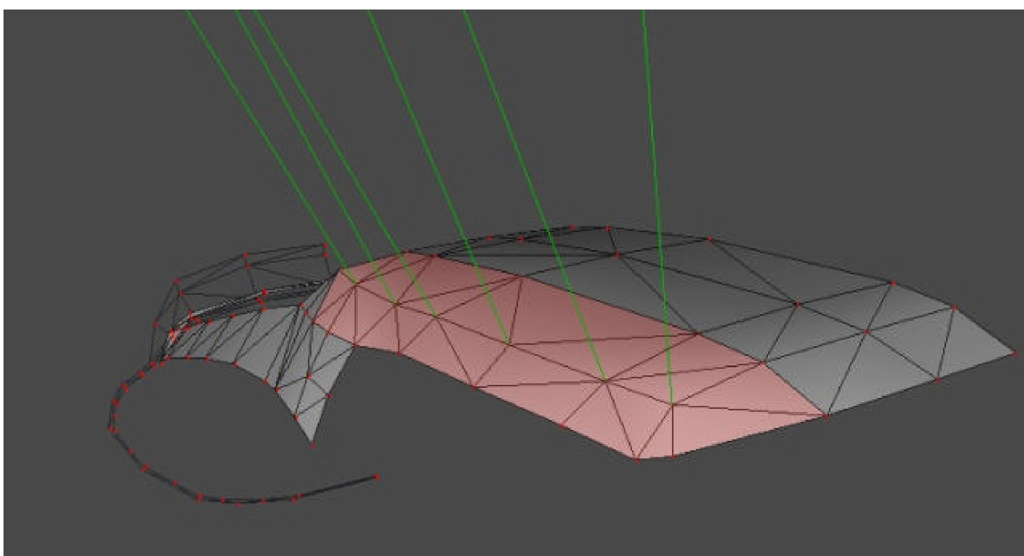
Creating 3d models has a very specific purpose. We want to create 3d representations of real (or imagined) things that look very real. Think about some of the aspects of reality that we would like to duplicate. First of all, we would like to create a surface in 3 dimensions that duplicates our real-life object. Next, we would like that surface to have the properties, such as reflection, and specularity. We also want that surface to have the correct texturing, or visible properties and patterns that the real object has. But all of this would not be convincing if your model was not *shaded*. Shading is something we may take for granted, but it is paramount to creating good 3d models. Shading is accomplished in 3d graphics by means of normals.

What is a Normal?

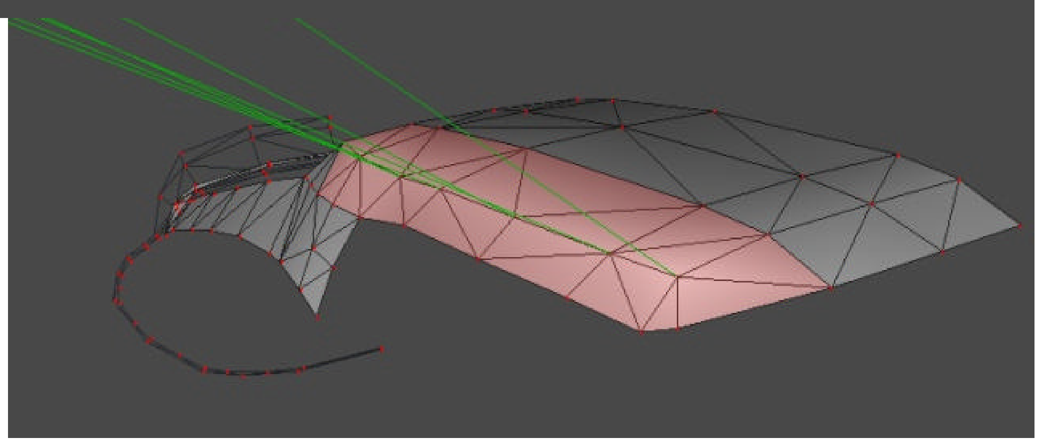
So this leaves us asking: "What on earth is a Normal? I'd rather be normal than abnormal!" Well, by now we should be used to all these imaginary constructs that are quite foreign to non-modelers. A normal is really a vector (a line that possess both length and direction). Every vertex has a normal vector, and this vector will determine how light from your 3d scene will reflect off the vertex, and the surfaces it is connected to. Normal vectors also affect the reflections on your model. Like I said, a vector has both direction and length. For example, speed is not a vector, because it has value but no direction. Acceleration is, however, a vector quantity, because it has a value, and a direction. So let's see what the length and direction of normals does to a mesh.

Direction

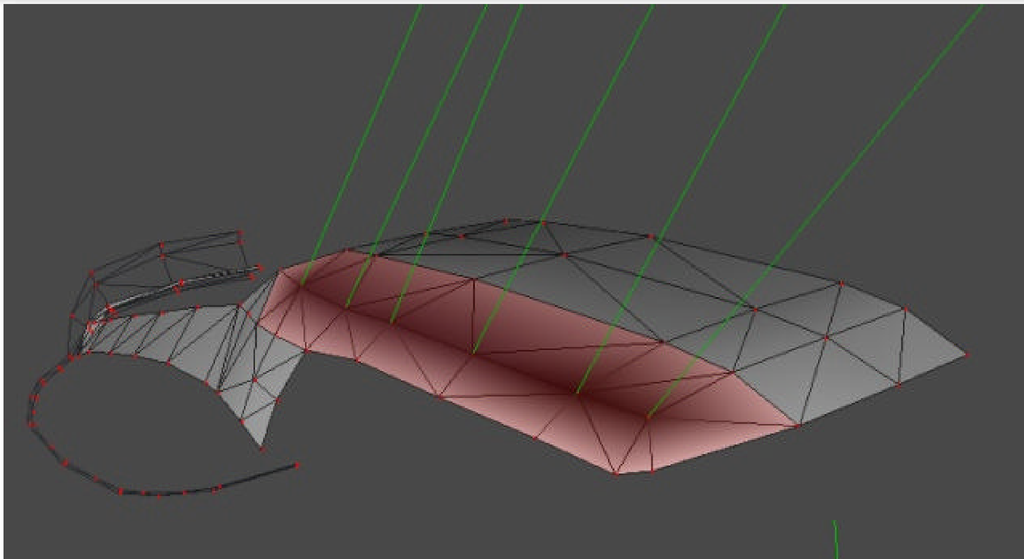
For lack of a more technical explanation, a normal's direction will affect how light reflects off of its related vertex. A vertex is often connected to quite a few faces, so it affects their lighting, as well. In the following two pictures, notice the direction of the Normals (the green lines) and notice the effect that rotating them has.



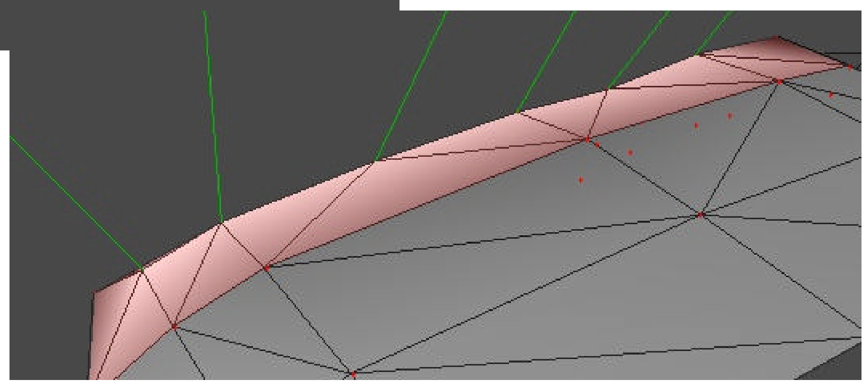
Notice the direction of the normals. And notice that the shading, or light reflection appears to be evenly distributed. Notice what happens when we rotate the normals outward:



In this instance, the surface reflects more light outward, in the new direction of the normals. On the next page, we'll rotate the normals in the other direction.



Now, the normals reflect light in the opposite direction for this line of vertices. Look at the model from the opposite side:

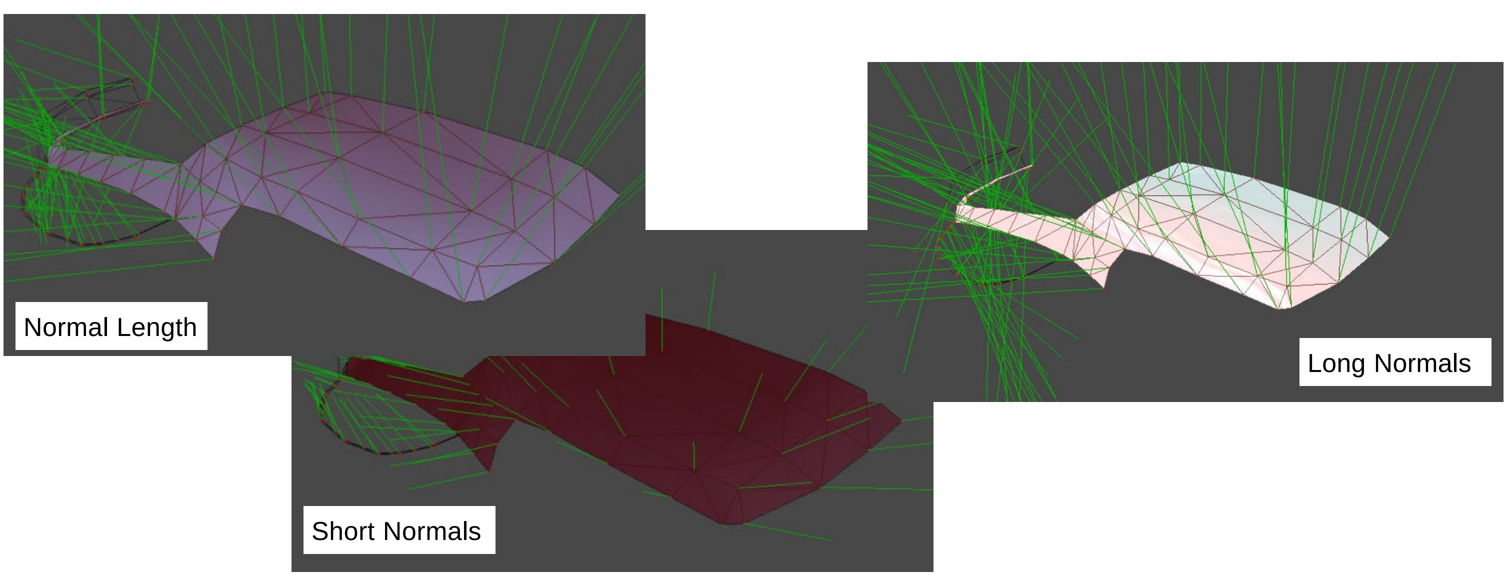


Now, that same strip of vertices is brightly lit, because we are now viewing the mesh from the angle that the normals are pointing to. This shows us a very important point. Normals may lighten or darken a mesh based on which angle we view it from.

If we were to sum it up into one sentence, we could say: “Normals’ direction indicates the direction that light will be reflected from a surface.” This is like a Sacred Rule of Normals Theory. We will see more of these throughout the chapter. As far as the reflective effects of Normals direction is concerned, the direction of normals tends to bend and distort environment mapping. it is very hard to illustrate these effects, however, suffice it to say, that if you get the shading correct using normals direction, then you will probably have gotten the reflections correct.

Length

With an acceleration vector, the longer the vector, the faster the acceleration. Similarly, the length of a normal is indicative of how much light is reflected in the indicated direction. With longer normals, you get brighter shading. With shorter normals, you get slightly darker shading, but more importantly, the differences between normals directions is less severe. When you have long normals, the difference between normal A pointing in one direction and the adjacent normal B pointing in another is great, and will have strong effects on the polys in between those two vertices. With shorter normals, that difference is not so pronounced. Take a look:



Normal Length

Short Normals

Long Normals

So from these three pictures we get two more Sacred Rules of Normals Theory:

1. The length of normals affects how brightly (or darkly) shaded a mesh is.
2. The length of normals affects how strong reflections are on the car's surface.

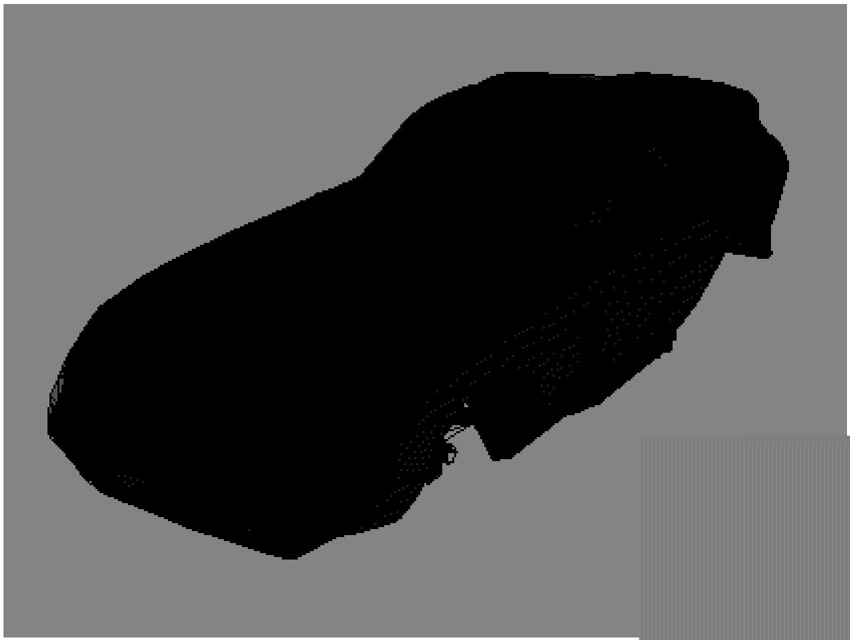
These rules are great, but how do we make them work for us? Let's talk about Manipulating Normals

Manipulating Normals

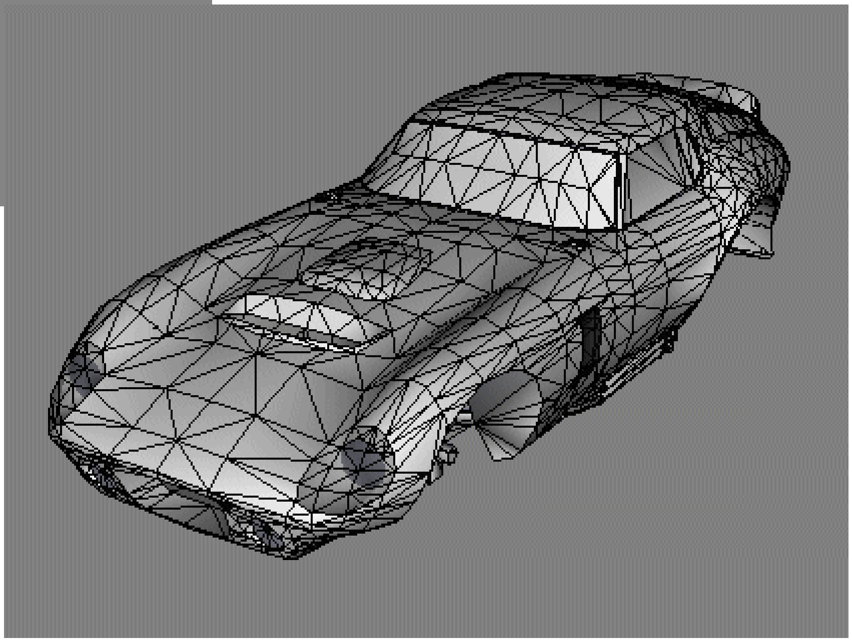
In all likelihood, your normals will not just fall right into place. More than 90% of the time, a mesh will need some manipulation of normals just to look right. This is because of the way normals are initially calculated. First let's go over the different ways you can automatically change normals on a car.

Calculation

The primary means of generating normals' positions is with *calculation.* Calculation basically considers all the vertices and calculates what their normals will be based upon their position and the angles from that vertex to each surrounding vertex. It then generates a normal that is the bisector of all those angles. It is smack-dab in the middle of all the angles from the said vertex to each surrounding vertex. We will see why these angles may be a problem in the future. When you apply the calculation to all the objects in your mesh, it calculates each vertex the same way. There is an option in Z-Modeler that recalculates the normals of any element that is modified. This means that as you move a vertex, it calculates the change in the normal as it moves based upon the difference in the angles. This feature keeps your mesh looking relatively usual while creating it. I have this feature turned off, however, because sometimes I don't *want* the normal to change as I modify things, because of some advanced normals techniques which we will learn later on. For now, I suggest you turn it off too. It is also illustrative for my next set of pictures. When you create a vertex with that feature off, the normal will not have any length at all. The surface will not be lit. It's similar to scaling the normals all the way down. So this is what an unlit mesh looks like:



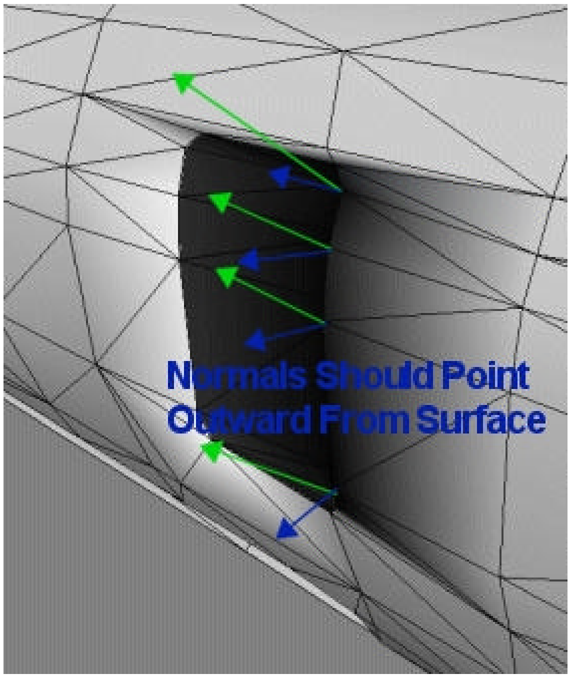
However, when you apply normals calculation:



You can see all the normals are generated correctly. There is shading and lighting. That's all we need, right? Well . . . .

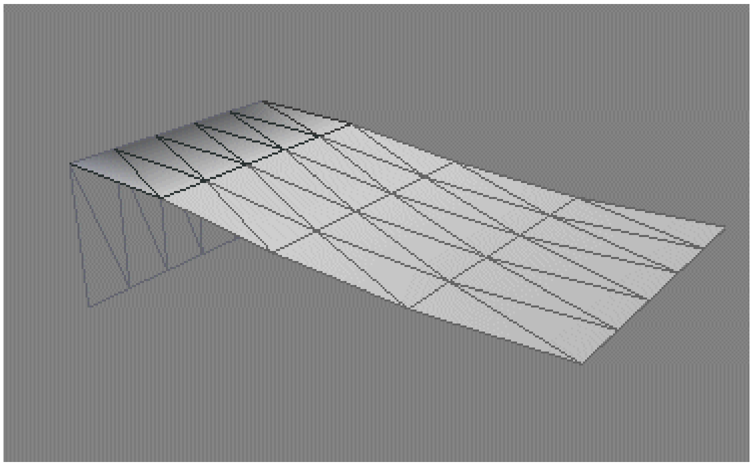
Problems with Normals Calculation

All is not well in paradise, however. Take a look at my Shelby Daytona Coupe:



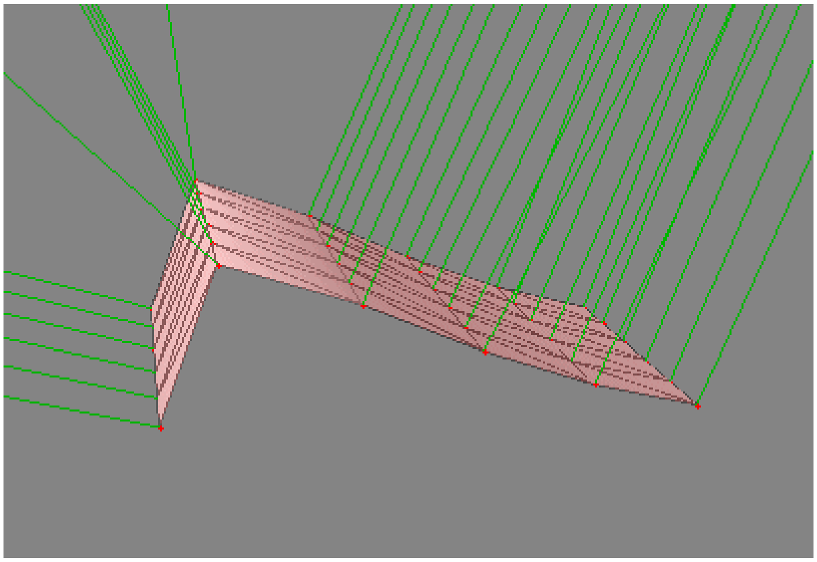
What's happened here? If we looked at the normals, we would see the vectors pointing, not outward from the car's surface, but rather, pointing backwards, away from the camera.

Remember our definition of Vertex Calculation? We said that "Calculation . . . considers all the vertices and calculates what their normals will be based upon their position *and the angles from that vertex to each surrounding vertex."* This last clause is the most important key to understanding normals. Normals will be calculated based upon the vertices' relationships to the surrounding vertices. Some times this is beneficial, but sometimes it is not. Take a look at this test mesh.



You can see, that the normals on the part of the mesh that is congruous are smooth and good. However, the sharp edge at the end is giving us problems. Let's take a look at why.

We see in this picture that the normals for each row of verts projects outward from the row, and is affected by the relative angles of the surrounding rows of vertices. The last row, where we have the 90-degree corner, does the same. The angles of the surrounding rows of verts for this row, are quite large. On the one hand, the row to the right is directly to the side of our row, but the row to the left is almost directly below our row. When calculating normals for this mesh, Z-Modeler has split the difference of these angles and has projected the normals for the corner row directly outward, normal to both the left and right surrounding rows.



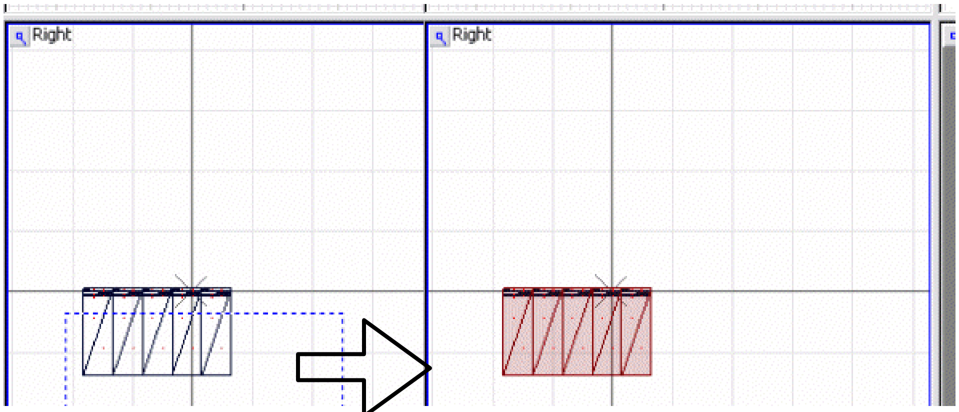
This is not a pleasant situation, primarily because when you view either extreme of the mesh, there is a blackened portion of the model. Let's discuss how to avoid this kind of Normal discrepancy.

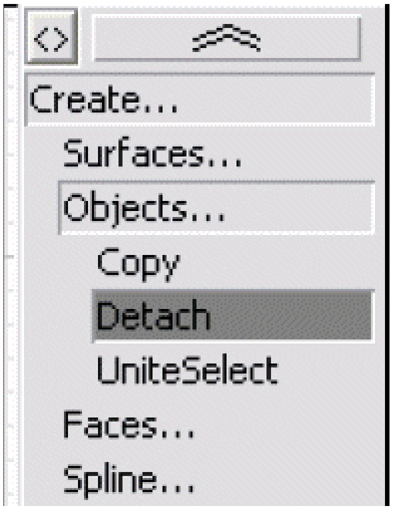
Detaching Faces

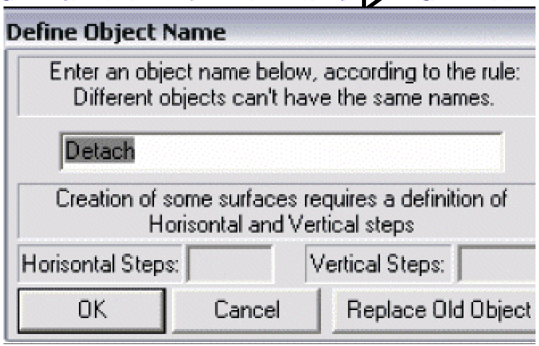
By analyzing the situation above, we can quickly see that the problem arises from the two sets of *faces* sharing the same normals. By this, I mean that the faces on both sides of the corner shared a set of vertices, and thus, a set of normals. The two sets of faces could not be shaded correctly because they both demanded sets of normals that could not be reconciled. The result was the incorrect shading above.

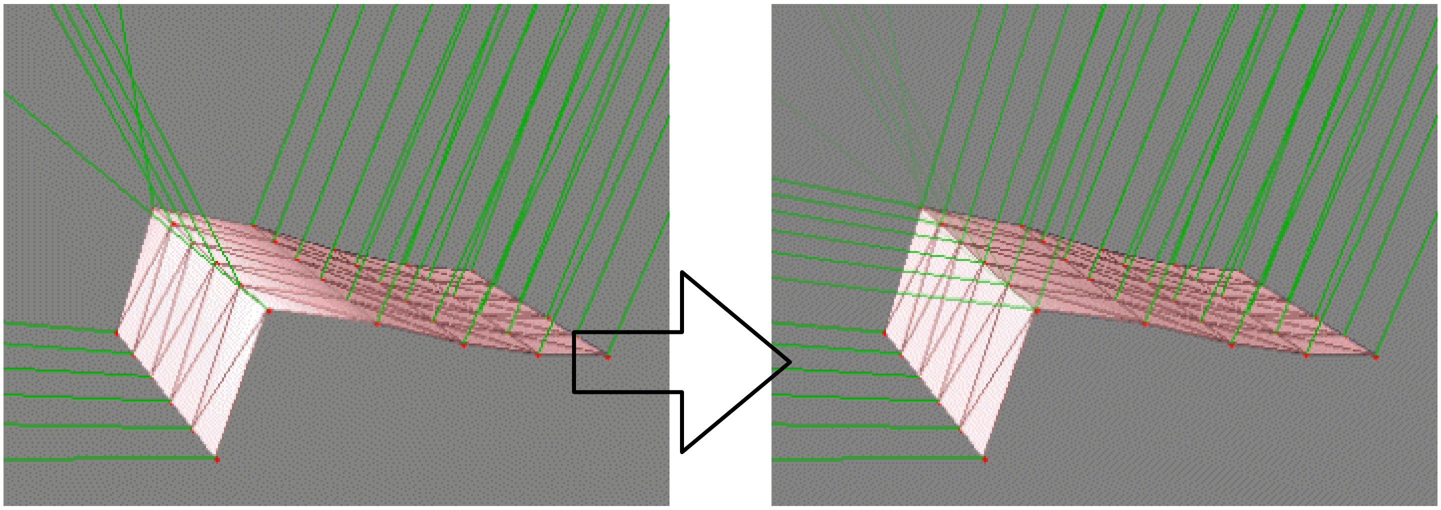
What if we could somehow split those verts, though, so that each set of faces could have its way? Well, to do this, the faces

can't share the same vertices. We need to detach one set of faces from the other. This way, they will not share any verts, but they will be adjacent, and for all intents and purposes, connected. How do you know which faces to detach? In this case, it's easy. The faces that point "downward" in the above picture are only attached to the mesh in one place, so they can be detached, and serve our purposes without any extraneous effects. Fortunately, Z-Modeler provides us an easy way to do this. Let's take a look:



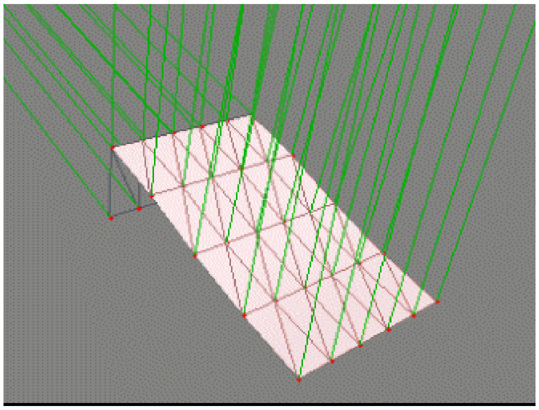






Let's see these steps in text:

1. Switch to the faces level of the part which contains the faces to be detached.
2. Select those faces which you wish to detach and put into their own object.
3. Be sure that no other faces are selected, this can have very bad consequences.
4. Activate Create-->Objects-->Detach
5. Activate SEL mode.
6. Click in a view-port, but be sure not to drag at all. If you do drag, you will pull the faces away from the object.
7. Enter a name for your new object
8. Click OK.



The detached faces will be their own object, and to see the results of what you've done, recalculate the normals of the two objects. The animation above shows you how it will look. Notice in the last frames how sharp the edge is between the two sides. That is the characteristic of using Detach. A rule of thumb is "Use Detach where you want a sharp line." We see this is true, viewing the results of our little experiment.

Using detach judiciously will give you some of the best results with Normals. However, sometimes you may find that Calculated normals are too "harsh". What can be done?

Projection

On low-poly models, normals are often spaced too widely apart, and often they represent too much change in angle from one vert to another to make smooth shading and reflections. The best way to alleviate this is to make high-poly models, but in the absence of this possibility, you can use Normals Projection. Normals Projection can be applied to verts or objects, or selections thereof. According to Z-Modeler's help system, the Normals Projection tool: "projects normals out from the axis center. . . . The direction depends upon the modify axis. To perform 3D projection you should hold SHIFT when clicking. To make a parallel projection you should hold down CTRL while clicking." What does that mean?

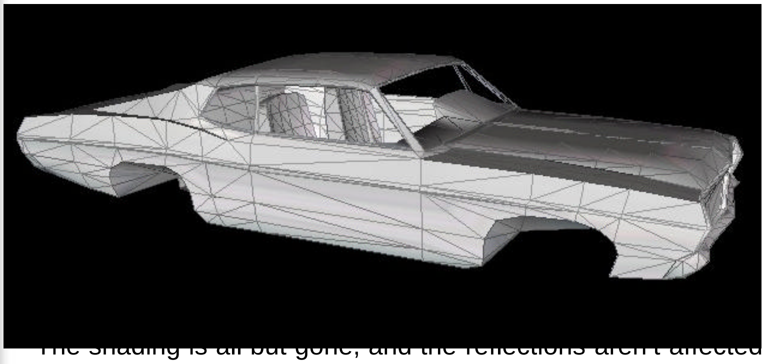
It means that Projection acts like wind, "blowing" the normals in one direction, as a stiff breeze would blow grass. This causes them to have a very different effect. In fact, it can be very undesirable. This is because the normals now point in a direction that may not be outward from the face. However, before we discuss how to make Projection look good, let's talk about how it works.

If you look at your model with all it's normals showing, they look a bit like fur. Projection acts like a hand smoothing that fur in all directions outward from the center. Which direction it smooths the fur, however, is dependant on which view port you click in. If you click on the top of the model, all the normals will be spread out from the top. If you click from the side, they will all be spread out from the side view, and so on. These differences have subtle effects. Two other advanced options are available too. By holding SHIFT while projecting normals from any view port, it is the same as having clicked in all view ports at once. It spreads the normals more evenly. The effect it has on the shading and reflections is dramatic. Projection in this manner will lessen the effect of shading on the car. By projecting the shading outward from the absolute center, instead of the center of each vert, the shading of the car depends on your viewing angle to it, not on the angles of each vert to one another. The effect this has on reflections is also apparent. Instead of the reflections looking "stretched" from one vertex to another, the reflection map looks "painted" over the entire surface, like a sheet draped over the car.

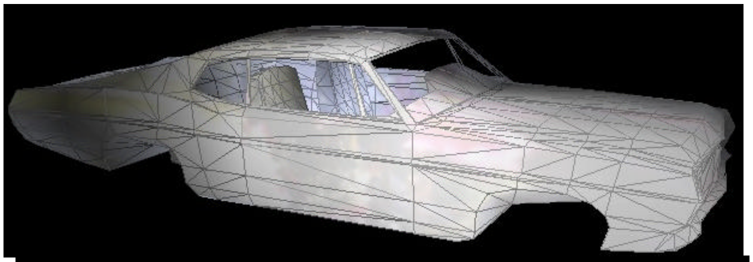
As we've mentioned, the projection method is not always the best one. However it's benefits can be utilized while minimizing its faults. The secret lies in blending different methods of Normals calculation.

Mixing Calculation and Projection

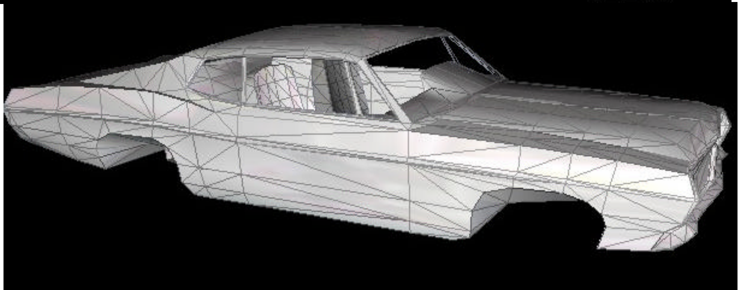
While Calculation or Projection may not be universally appropriate, we can blend the two methods together to get a combination effect which is quite good looking. Let's take a look at an example. Here is a Galaxie mesh which I made a long time ago. It's good, and as you can see, certain parts have been detached, to make sharp edges.



However, I feel that the edges are too sharp. Also, you'll notice that the reflections are distorted. If I project then normals, I don't like that either:



The shading is all but gone, and the reflections aren't affected by the car's shape at all. So, let's try 75% Calculation, 25% Projection.



(you may have to zoom in to see the images fully)

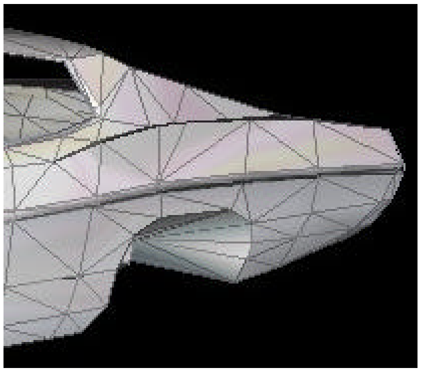
Much better, don't you agree? The shading now is defined, but it is slightly darker away from the camera. The reflections are also less distorted. Before you get all jealous, let's go over how I did that. If you read the earlier chapters, you might remember that we talked about the Numeric Bar #1. This is a valuable little tool, and it's good to have it open all the time. If you have it open, and activate Surfaces-->Normals-->Projection, you'll see it change from saying "N/A", to saying "% Applyment", which indicates how much the projection is applied to the surface at a time. You can project once at a certain percentage, and then do it again to get a progressively more "Projected" look.

So, first, activate Calculation, select all objects, activate SEL mode, and click in any view. The model will have fully projected normals. Now, activate Projection, and click on the down arrow in Numeric Bar 1 until it reads "25% applyment". Then, hold shift, and click in any view. You'll have 25% projected normals!

Combining Methods

I think the biggest lesson to learn here is to find the right balance of all these methods to make your shading realistic. In general, I try not to detach too many different areas. This tends to make the car look very "blocky" and overly sharp. Get good photos of the car, and note the places where sharp lines are appropriate. Also, don't be afraid of a reflections that are a little bit stretched and distorted. Note how a real car's surface distorts reflections in the real world. View your mesh with an environment map with text in it, and see if the text is really distorted or just stretched, but readable. If you can read it or tell that it's text, you're probably OK.

Another big help when making normals is to remember that you can calculate and/or project normals on only one part, or on a group of verts, or just one vert, if you want to. Experimenting with methods on different parts of the car can help you create edges that are more than just hard or soft. Notice this one:



Here I decided that on the ridge along the side of the Galaxie I would fully project the normals of the outer line of verts, because I wanted the bottom side of the ridge to be darkened. So after calculating, applying overall 20% projection, I applied 100% projection to the outer line. I really like the result. This didn't have any adverse consequences because the line I projected is surrounded on top and bottom by a line of polys, so the shading did not "bleed" into the rest of the car.

One final thing to mention is the bonus tool you receive when you register Z-Modeler. Oleg showed me how to use this one in passing, and it really helps out when quickly doing normals in the middle of development of a car, and even when you're finalizing your normals at the end. The tool is the Normals Tuner. This handy little tool allows you to drag sliders and mix Calculated normals, Projected normals, and the current normals for a mesh.

To use it, select the objects you wish to tune, activate SEL mode, and click on Tools-->Adjust Normals-->Normals Tuner. This brings up a little window, with sliders for Current, Calculated, Projected, and the ability to select the modification axis. As we've seen, normals projection is best on all axes, so check the "Lock Axis" box. Then, drag the sliders for different effects. You can also experiment with unlocking the axis and trying different modification axes.

Now that we've discussed normals, I must reiterate that learning is an ongoing process. I'm still learning great new techniques about normals, and you can discover them too. I'm open to new ideas and I welcome any good techniques you may have come across. Normals don't have any hard-and-fast rules. Remember to be creative, and never be satisfied, and you'll come out with some great normals for your model.